Using Segmentation Constraints in an Implicit Segmentation Scheme for On-line Word Recognition

Émilie Poisson Caillault, Christian Viard-Gaudin

Using Segmentation Constraints in an Implicit Segmentation Scheme for On-line Word Recognition

Émilie Caillault and Christian Viard-Gaudin
IRCCYN UMR CNRS 6597
École polytechnique de l'université de Nantes
Rue Christian Pauc - 44306 Nantes Cedex 3, France
{emilie.caillault ; christian.viard-gaudin}@univ-nantes.fr

Abstract

In this paper, we propose the introduction of a parametric weighting function to constrain the segmentation path which governs the training of a hybrid neuro-markovian scheme for an on-line word recognition system. Due to the parametric properties of this function, it is possible to modulate the constraints from imposing a strict balanced path with the same duration for every states to a totally free segmentation path. So far, during the initialization step of the training, when the neural network has little ability to correctly segment the word into its basic constituents, the constraints will be activated, and then, they will be relaxed to allow more flexibility to the segmentation-recognition process. Recognition experiments on the Ironoff database demonstrated that the proposed method allows to increase the word recognition rate when compared to a totally unconstrained segmentation training.

Keywords: control of implicit segmentation, online cursive words recognition, discriminant criteria, TDNN, dynamic programming

1. Introduction

In recent years there has been a significant body of work concerning online handwriting recognition systems [6]. Many of these works are restricted to solve the recognition problem for isolated digits or characters. It is much more complex to tackle the problem of unconstrained words. This is why, to alleviate the segmentation problem, one can impose specifics constraints, such as using a script style [1, 5]. Systems being able to process unconstrained styles for writer independent recognition still require much research effort to design very performing systems with limited computational and memory resources. The key idea to circumvent the difficulties involved with unconstrained word recognition is to combine the segmentation and recognition steps. Segmentation-recognition methods first loosely segment (over-segment) words into graphemes that ideally consist of either characters or parts of characters, and use dynamic programming techniques together with a lexicon to find the definitive segmentation as well as the best word hypotheses. Many systems use HMMs [7] to model sub-word units (characters) and the Viterbi algorithm to find the best match between a sequence of observations and the models [3, 4, 8, 10].

Analytical methods employ segmentation-based recognition strategies where the segmentation can be explicit [3, 5, 10] or implicit [4, 8]. In the case of explicit approaches, the idea is to use heuristic techniques to find candidates cuts of the words, and then to use the recognizer to score the alternative segmentations thereby generated. Several systems have been developed on such an approach, in the case of off-line handwriting cuts are generally based on the lower points of the upper profile while for on-line handwriting, pen-up, crossing points, and down-stroke extremities are used to generate the set of potential segmentation cuts.

Alternatively, with an implicit segmentation scheme, the idea is to sweep a recognizer at all possible locations across the whole word signal. With this technique, no segmentation heuristics are required since the system essentially examines all the possible segmentations of the input.

The system used in this paper [2] is an analytical implicit segmentation-recognition system, which combines a TDNN (Time Delay Neural Network) which encodes the likelihood of finding character of a particular class at the corresponding location of the input, and a dynamic programming scheme, which is used to extract the best possible label sequence from this vector sequence. The difficulties involved with such a system rely on the training procedure. We have defined a global training approach, working directly at the word level without any specific training at the character level, as it is generally proposed [4, 8, 9]. However, we introduce in this paper a parametric weighting function which is used to constrain the Viterbi segmentation path in such a way that it is possible to monitor the length of the segmentation with respect to the different letters of the word. A measure defining an average segmentation homogeneity criteria is also proposed, and our experiments show that the proposed method allows to increase the word recognition rate when compared to a totally unconstrained segmentation training.

2. Online cursive word recognition

In this section we briefly introduce an overview of the system used for the recognition of online cursive handwritten words. It is presented in more details in [2].

2.1. Overview of the global system

Figure 1 gives an overview of the complete on-line recognition system. It is based on an analytic approach with an implicit segmentation and a global word-level training.

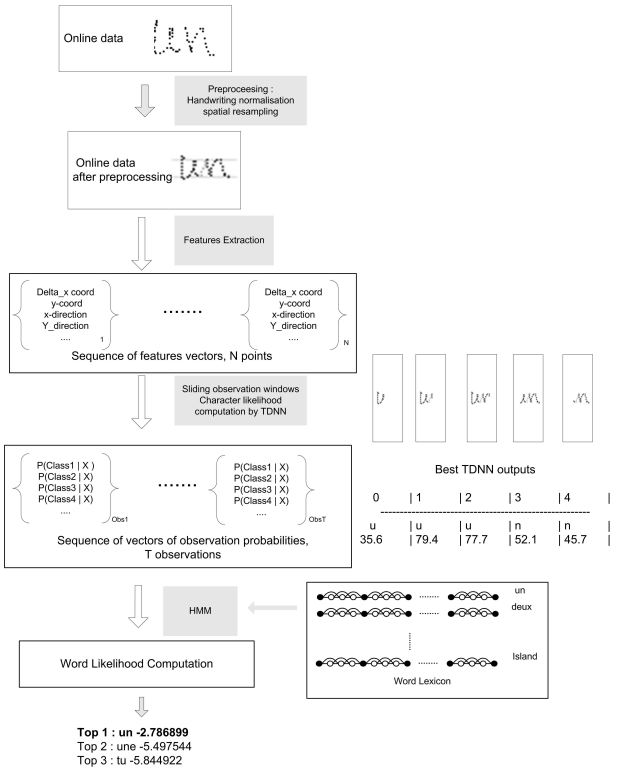

Figure 1. Overview of the on-line cursive words recognition system

Thus, it allows to handle dynamic lexicon, and no additional training is required to add new entries in the lexicon. Some pre-processing steps are first introduced in order to normalize the input signal, specifically with respect to size, baseline orientation and writing speed. From these normalized data, a feature-vector frame is derived, $X_{1,N} = (x_1, \ldots, x_N)$, where x_i describes the ith point of the input signal. It will be the input of the NN-HMM learning machine. The role of the NN in this hybrid system is to provide observation probabilities for the sequence of observations, whereas the HMM is used to model the sequence of observations and to compute word likelihood, based on the lexicon. As a NN, we have used in a previous work [10] a standard multi-layer perceptron (MLP) with an explicit multi-segmentation scheme, whereas in this work, we have privileged a Ms-TDNN [4] with no explicit segmentation at the character level but a regular scan of the input signal $X_{1,N}$ to produce the probability observation $O_{1,T}$. For each entry in the lexicon, a HMM-Word model is constructed dynamically by concatenating letter HMMs (66 characters: lowercases, uppercases, accents and symbols). The characters are modeled with one or several states. So in our TDNN, there are as many outputs as the total number of states. For example, if we describe all characters with three states, the number of outputs is 198 (66 letters $\times$ 3 states). Observation probabilities in each emitting state of the basic HMMs are computed by the NN. Usually, transition probabilities model the duration of the letters, actually, as we will transfer this modeling to the weighting function which will be introduce latter (section 4), we assume the same duration for every letter, all transition probabilities are set to 1 and are not modified during training. Hence, the likelihood for each word in the lexicon is computed by multiplying the observation probabilities over the best path through the graph using the Viterbi algorithm, see figure 2. The word HMM with the highest probability is the top one recognition candidate.

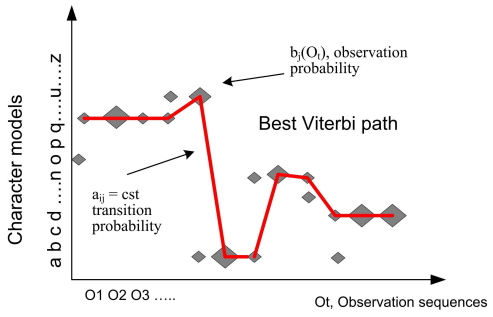

Figure 2. Recognition treillis and Dynamic programming for the french word "quand"

2.2. Word-Level Training and Results

Training such a system could be imagined either at the character level, or directly at the word level. The character level requires to be able to label the word database at this character level, usually using a post-labeling with the Viterbi algorithm, and to iterate several cycles of training/recognition/labeling to increase the overall performances. There are some difficulties involved with such a scheme. One is to bootstrap the system with an initial labeling, a second problem is to transform, the posterior probabilities estimated by the ANN into scaled likelihood, a third problem is to deal with inputs that have not been encountered during the training because they do not correspond to any actual character. In order to simplify the training process and to improve the word recognition rate, we proposed in the paper [2] a global training of the hybrid system at the word level. In that case, there is no training explicitly at the character level (no previous or hand labeled character database) but an optimization of the network to satisfy an objective function defined at the global word level. This objective function combines dis-

criminant criterion at word-level and at the character level, it is defined by the following relation (Eq. (1)):

$$\begin{aligned} L_G &= (1+\epsilon)\log P(O|\lambda_{trueHMM}) \\ &\quad -\beta(1-\alpha)\log P(O|\lambda_{bestHMM}) \\ &\quad -\beta\alpha\log P(O|\lambda_{bestTDNN}) \end{aligned} \quad (1)$$

where α, β and ϵ being mixture parameters belonging to [0..1]. With $\beta = \epsilon = 0$, we get the bare Maximum of Likelihood (ML) function, whereas with $\beta = 1$ we introduce a discrimination training that takes into account either only the best word-HMM if $\alpha = 0$ (that corresponds to a simplified Maximum Mutual Information criterion - MMIs), or only the best-TDNN classes if $\alpha = 1$. An intermediate α value interpolates between these two situations.

The online words available in the IRONOFF database [11] were used in these experiments. The whole training set of words (20 898 words representing 197 different labels) is used for training and a separate set of 10 448 words is used to test the system. We train the system with different models for a character : one state, two states, and three states. For each configuration, the number of outputs of the TDNN increases with the number of states S per letter (66×S output neurons). Table 1 shows the recognition rates on the test set and we note the interest to expand the number of states. In all the cases the generic criterion achieves better results, and the 3-state model allows a 44% error rate reduction with respect to the basic 1-state model.

Table 1. Recognition rates (%) for one/two/three state per letter on IRONOFF word database.

Criterion	MLE	MMIs	MMIs + MLE	Mixed
Parameters	$\epsilon=0$ $\beta=0$ $\alpha=0$	$\epsilon=0$ $\beta=1$ $\alpha=0$	$\epsilon=1$ $\beta=1$ $\alpha=0$	$\epsilon=1$ $\beta=1$ $\alpha=0.5$
1 state	77.43	83.82	86.34	87.09
2 states	80.87	87.46	88.22	90.51
3 states	84.69	90.57	92.01	92.78
Relative improvement w.r.t 1-state	32	41	41	44

3. Weakness of the proposed system

The proposed system achieves quite reasonable performances when using the mixed criteria with 3 states per letter, but nevertheless, it is far for being perfect. In order to understand its main limitations, we have analyzed the principal causes of errors of the system. Illustrated on Figure 3, they are characterized mainly by:

- An incorrect detection of the references lines
- The absence of specific processing for diacritic marks
- A bad discrimination of lexically close words
- A deficiency in the segmentation/recognition process

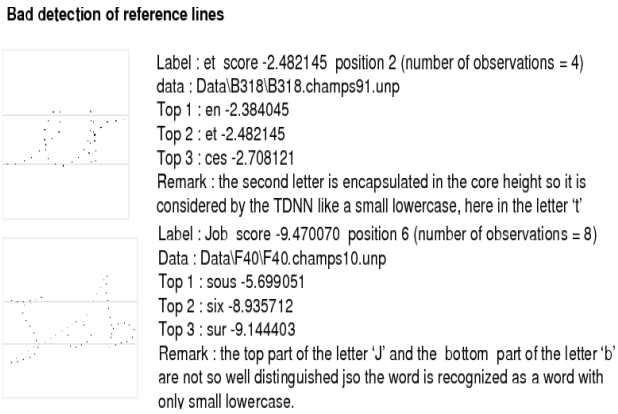

Figure 3. Illustration of recognition errors

The first two points should be taken into account in the pre-processing stages of the system. Figure 4 displays the recognitions rates according to the word length for the MMI-MLE criterion. Obviously the short words (up to 3 letters) are poorly recognized. This is mainly due the pre-processing steps, which are applied in the same way whatever the length of the word are, the core line extraction being specifically sensitive to the word length. It is more difficult to detect reference lines with a limited number of extrema. Figure 3 illustrates two recognition errors mainly

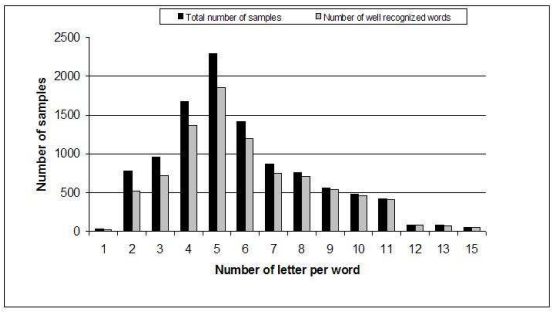

Figure 4. Evaluation of the robustness of the system according to the number fo letters per word

due to imprecise core lines extraction: in the word "et" the upward extension of the second letter is very small, hence leading to the recognition of the word "en", while with the second example, it is the first letter which does not extend significantly above the core lines preventing to recognize the correct label "Job" (the word "job" could not be proposed because it is not in the lexicon).

The two last causes are more directly linked to the essence of the proposed method. With the multi-state representation, we have been able to absorb the variability and the noises due to the ligature between letters in a word. This has lead to substantial improvements as dis-

played in Table 1. Another problem is the initialization of the training procedure of the TDNN in the segmentation/recognition process. The problem is that, as long as the TDNN is not able to correctly push up the actual character corresponding to the center position of the observation window, the Viterbi sequence built on the true word model could be very unbalanced, and will not correspond to the real sequence of letters. To reduce the effect of non regular segmentation, it would be advisable to help the network to learn the good way to segment a word. This initial step will add constraints to determine the optimal segmentation path. These constraints will be then relaxed progressively until disappearing.

To illustrate this problem, let us examine an example of sequence with the word "de" and let us assume that it is scanned by the TDNN with 10 observations. The segmentation problem is to distribute these 10 observations, the first ones being assigned to the states of the letter 'd', the last ones to those of letter 'e'. With a random initialization of the TDNN, all outputs are equally probable whatever the input is, and consequently the optimal path computed by the Viterbi algorithm could be as well any possible paths from "deeeeeeeee" to "ddddddddde". Of course, this segmentation and the associated correction affect the training capability of the system. If we consider that the Viterbi path is for instance "ddeeeeeeee", then the gradient matrix used for the back propagation in the TDNN is given by Table 2.

Table 2. Gradient matrix for a MLE training of the word "de"

Letter	O_1	O_2	O_3	O_4	O_5	O_6	...	O_{10}	Σ
d	1	1	0	0	0	0	...	0	2
e	0	0	1	1	1	1	...	1	8
Others	0	0	0	0	0	0	0	0	0

In that case, the letter 'e' is corrected 4 times more than the letter 'd'. This unbalanced correction is not satisfying since there is little chance that it corresponds to the real situation. It would be desirable to compensate for the deficiency of the NN in this initialization stage to clearly identify the actual character. To obtain a more balance segmentation, it would be necessary to obtain for the distribution of the probability of the various possible segmentation paths, a function as presented in Figure 5.

With such a function, if all TDNN outputs have about the same probabilities then the selected path will distribute uniformly the observations to the corresponding letters. With this example, the returned path will be "dddddeeeee". However, if there is a strong evidence for a letter in a given position, it has to be taken into account.

To address this problem two current solutions are proposed in the literature. They require a training with a character database: the first one with a duration model for HMM sub-unit and the second one with manual segmentation constraints. The first and most common way to overcome this segmentation problem is to use a duration model resulting in a specific topology for the Markov model. In Penacee [8] the training consists in 3 steps. The first two steps correspond to the training of on isolated characters and the training of transitions (i.e. not valid characters). The last step based on a ML criterion used a duration model for each HMM character category where the transition probabilities follow a Poisson distribution. The second method consists to do a first training with a hand labeled character database with the constraint of same number of passages in each state in the character model, see Remus [13] or Npen++ [4]. This constraint is then relaxed on the characters database and finally a ML training on a word database is performed.

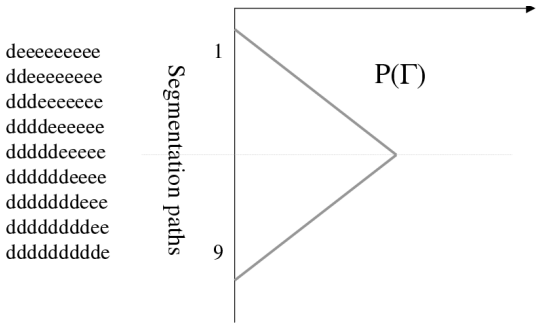

Figure 5. Desired form of the likelihood distribution on the different segmentation paths

We propose another method that does not modify the structure of the Markov models (transitions equals to one) and does not use a character database but that integrates a duration model directly in the emission probabilities. We can note that a training with a discrimination directed by the TDNN outputs ($\alpha > 0$ in the objective function Eq. (1)) allows us to do an initialization of the TDNN and avoid to use a character database.

4. Implicit segmentation control

To control the segmentation in order that it does not rely solely on the TDNN outputs but also on some a priori knowledge concerning handwriting, we apply a weighting function, which is defined by a simple parametric triangle function noted f_T centered on each letter of the word label, see (Eq. (2)).

$$\begin{aligned} f_T(t,B,E,H) &= 1_{t\in[B,B+S/2]}(1+2(t-B)(H-1)/S \\ &\quad +1_{t\in]B+S/2,E]}(1-2(t-E)(H-1)/S \\ Outputs(t,l) &= min(Outputs(t,l)*f_T(B,E,H,t);1) \end{aligned} \tag{2}$$

with
B = First observation position for the concerned letter
E = Last observation position for the concerned letter
H = Height of the triangle function f_T
$S = E - B$ = letter size

By doing this, we keep the same topology for the

whole hybrid system with always unitary transitions between states, the duration model being incorporated within the emission probabilities. The width of the triangle function is set to the average number of observations per letter (number of observations for the word/number of letters) and its height H allows to monitor the strength of the segmentation constraint. Figure 6 illustrates the shape of the TDNN output weighting function for one considered letter with two different H values. When H is more than 100% then the central points have the same weight due to the clamping threshold, allowing to move the segmentation point inside this range without penalty. Conversely, when H is less than 100%, only one optimal path is possible, the one that associates the average number of observations per letter to every letter. In that case, and for the previous example, the segmentation will effectively be "dddddeeeee".

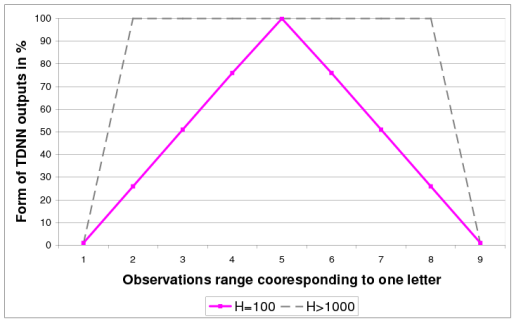

Figure 6. Profile of the weighting function centered on the observations associated with a given letter

In the case of the example "de" and a triangle function with $H = 100$, the weighting functions for the two letters 'd' and 'e' are displayed in Figure 7, the values being bounded by 1 and 100%. Only the reference path will be constrained, i.e. the path of the true label, by weighting the observations probabilities corresponding to the letters belonging to this label at their respective location.

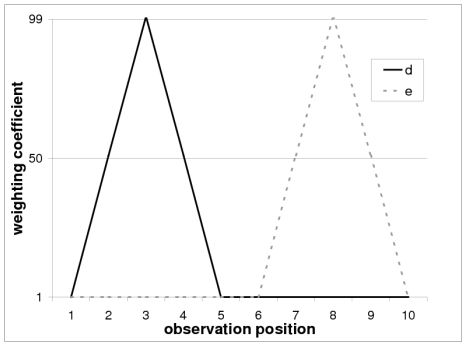

Figure 7. Weighting of the TDNN outputs for the word "de"

In the case of a multi-state TDNN all the states of the same letter are weighted by the same function.

Now that we have defined this weighting function to constrain the segmentation, it would be interesting to define a measurement to evaluate its effectiveness. For that purpose, we introduce the rate of regularity of segmentation for a word, called ASR for Averaged Segmentation Rate (Eq. (3)); which is defined by the geometric mean of the ratios of the number of observations for a character over the number of observations awaited by character.

$$ASR = \frac{1}{N}\sum_{e=1}^{N}\left(\prod_{i=1}^{NLs(e)}\frac{nb\_obs(i)}{NLc(i)*T/NL(e)}\right) \quad (3)$$

with
e: index of the considered sample ;
T: number of observations scanned by the TDNN ;
N: (fixed or limited) number of training samples ;
NL(e): number of letters in the word label ;
NLs(e): number of sequences of different letters ;
i: index of a sequence of letters from 1 to NLs ;
NLc(i): number of the same consecutive letter ;
(ie for "bee" NL=3, NLs = 2, NLc(1)=1, NLc(2)=2) ;
and nb_obs(i): number of observations of the letter i in the optimal Viterbi path.

This ratio is equal to one in the case of a homogeneous distribution of the number of observations. The more distant the segmentation of the word will be from a regular distribution, the more this average ratio will tend towards 0. If one takes again the example of the word "de" to 10 observations, the table 3 below gives the rate of regularity of segmentation for the various possible segmentations of this word.

Table 3. Rate of regularity of segmentation for the various possible segmentations of the word "de".

NL(d)	1	2	3	4	5	6	7	8	9
NL(e)	9	8	7	6	5	4	3	2	1
ASR	0,36	0,64	0,84	0,96	1	0,96	0,84	0,64	0,36

This rate can be used to check the effectiveness of the constrained segmentation procedure, and could be used to relax this constraint once a reasonable rate has been obtained and/or a given number of iterations of the training database has been carried out, so that the TDNN has been trained enough to take over the training process.

5. Experimental Results

We have divided the training stage in two steps. The first step corresponds to the training with a MLE criterion constrained by a duration model for 20 epochs and the second step to train the system without segmentation constraint and with a discriminant criterion.

The different discriminant criteria introduced in section 2 have been tested, and Table 4 presents the corresponding results for two hmm topologies:1 state and 3 states per letter. In addition to the recognition rate, the relative improvement in the recognition rate has also been

indicated.

Table 4. Influence of a constrained segmentation for different criteria : recognition rate (RR %) and relative improvement w.r.t. unconstrained segmentation (no Duration Model)

Criterion	MLE	MMIs	MMIs+ MLE	Mixed
1s-TDNN	77.61	88.02	89.04	91.31
w.r.t no DM	0.79	25.9	19.7	32.6
3s-TDNN	85.10	92.43	93.38	94.81
w.r.t no DM	2.67	19.7	17.1	28.1

From Table 4, we observe that the performances with the ML criterion remain similar in the 1-state TDNN or 3-states TDNN with or without the constrained segmentation, while they progress significantly with the other discriminant criteria. For instance, there is a reduction of the word error rate of more than 28% with the 3-states model when using the mixed objective function, in that case the recognition rate reaches 94.8% instead of 92.8% without any duration model to constrain the segmentation.

It is interesting to compare these results with those presented in Table 5. It displays the values of the average segmentation rate, as defined above (Eq. (3)), according to the type of objective function used to train the hybrid, and with or without the segmentation constraint.

Table 5. Influence of a control of the implicit segmentation for different criteria : regularity rate of segmentation (ASR) with or without Duration Model (DM)

Criterion ASR	MLE	MMIs	MMIs+ MLE	Mixed
1s - no DM	0.260	0.290	0.306	0.423
1s -DM	0.273	0.663	0.503	0.498
3s - no DM	0.325	0.403	0.438	0.612
3s -DM	0.369	0.632	0.543	0.684

We can observe that the ASR values are highly correlated with the recognition results, and that the introduction of the duration model, which is only applied during the first 20 epochs, increases systematically the average segmentation ratio computed on the test database. With the MLE criterion, this effect is not very important, while it is much more significant with the other criteria. One possible explanation is that with the MLE criteria, once a word is in the top 1 position, it does not participate any more to the training process, even if the individual outputs produced by the TDNN are far for being optimal.

6. Conclusion and future works

We have pointed out in this paper the limitation encountered in the training of a hybrid word recognition system directly at the word level. There are some difficulties to bootstrap the system correctly without any hand labeled character database. The proposed solution uses an initialization stage where some constraints are added during the segmentation step, they are simply introduced as a weighting function which affects the emission probabilities computed by the TDNN. We show that it allows to obtain a segmentation of better quality and most importantly that the recognition results were increased. Up to now, we have manually switched from the controlled segmentation stage to the unconstrained one, but it would be possible to gradually loosen the constraints by modifying the H parameter of the weighting function, and to adapt it automatically with respect to the ASR value.

References

[1] E. Anquetil, H. Bouchereau, "integration of an Online Handwriting Recognition System in a Smart Phone Device", *in Proc. of 16th IAPR-International Conference on Pattern Recognition)*, Quebec, 2002, pp. 192-195.

[2] E. Caillault, C. Viard-Gaudin, "MS-TDNN with Global Discriminant Trainings", *Proc. of 8th International Conference on Document Analysis and Recognition*, Seoul, Corea, august 2005, pp. 856-860.

[3] W.T. Chen, P. Gader, "Word level discriminative training for handwritten word recognition", *in Proc. of 7th International Workshop on Frontiers of Handwriting Recognition*, ISBN 90-76942-01-3, 2000, pp. 393-402.

[4] S. Jaeger, S. Manke, J. Reichert, A. Waibel, "On-Line Handwriting Recognition: The NPen++ Recognizer", *International Journal on Document Analysis and Recognition*, 2000, volume 3, pp. 169-180.

[5] L. Oudot, L. Prevost, M. Milgram, "An activation-verification model for on-line handwriting texts recognition", *Proc. of 9th International Workshop on Frontiers of Handwriting Recognition*, Tokyo, Japon, 2004.

[6] R. Plamondon, S.N. Srihari, "On-Line and Off-line Handwriting Recognition: A Comprehensive Survey", *IEEE Transactions on PAMI*, 2000, Vol.22, No. 1, pp.63-84.

[7] L. R. Rabiner, "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition", *IEEE*, 1989, vol. 77, pp. 257-285.

[8] M. Schenkel, I. Guyon, D. Henderson. "On-line cursive script recognition using Time Delay Neural Networks and Hidden Markov Models". *Machine Vision and Applications, special issue on Cursive Script Recognition*, 1995, (8), pp. 215-223.

[9] G. Seni, R. Srihari, and N. Nasrabadi. "Large vocabulary recognition of on-line handwritten cursive words". *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 18(7), July 1996.

[10] Y.H. Tay, P.M. Lallican, et al., "An Analytical Handwritten Word Recognition System with Word-Level Discriminant Training", *Proc. of Sixth International Conference on Document Ananlysis and Recognition*, Seattle, Sept. 2001, pp. 726-730.

[11] C.Viard-Gaudin, P.-M. Lallican, et al..The Ireste ON/OFF (IRONOFF) Dual Handwriting Database. *Proc. of Fifth International Conference on Document Analysis and Recognition*, 1999, pp. 455-458.

[12] H. Weissman, M. Schenkel, I. Guyon, C. Nohl, D. Henderson, "Recognition-based segmentation of off-line run-on handprinted words: Input vs. output segmentation." *in Pattern Recognition*, 27 (3), 1994.

[13] Z. Wimmer, S. Garcia-Salicetti, A. Lifchitz , B. Dorizzi, P. Gallinari, T. Artires, REMUS , http://www-connex.lip6.fr/ lifchitz/Remus/ .